The Depth-Restricted Rectilinear Steiner Arborescence Problem is NP-complete

Jens Maßberg [*]

August 28, 2015

Abstract

In the rectilinear Steiner arborescence problem the task is to build a shortest rectilinear Steiner tree connecting a given root and a set of terminals which are placed in the plane such that all root-terminal-paths are shortest paths. This problem is known to be NP-hard.

In this paper we consider a more restricted version of this problem. In our case we have a depth restrictions $d(t) \in \mathbb{N}$ for every terminal t. We are looking for a shortest binary rectilinear Steiner arborescence such that each terminal t is at depth $d(t)$, that is, there are exactly $d(t)$ Steiner points on the unique root-t-path is exactly $d(t)$. We prove that even this restricted version is NP-hard.

keyword: Steiner arborescence, depth restrictions, NP-completeness, VLSI design, shallow light Steiner trees

1 Introduction

1.1 Problem Description

Let $T = \{t_1, \ldots, t_n\}$ be a set of terminals with positions $p(t) \in \mathbb{R}^2$ in the plane for all $t \in T$, a distinguished terminal $r = t_1$, which we call the *root*, with $p(r) = (0,0)$ and a function $d : T \setminus \{r\} \to \mathbb{N}$. A *Depth-Restricted Rectilinear Steiner Arborescence* is an arborescence A with root r, leaves $T \setminus \{r\}$ and an embedding $\pi : T \to \mathbb{R}^2$ in the plane such that

- each Steiner point, that is, each vertex of A that is not a terminal, has degree 3,

- each terminal $t \in T$ has degree 1,

- $\pi(t) = p(t)$ for all $t \in T$,

[*]Institut für Optimierung und Operations Research, University of Ulm, jens.massberg@uni-ulm.de

- the unique path P in A from r to $t \in T$ is a shortest path with respect to rectilinear distances, that is,

$$\sum_{(v,w) \in E(P)} ||\pi(v) - \pi(w)||_1 = ||p(t) - p(s)||_1$$

and

- for each $t \in T$ the number of internal vertices on the unique path from r to t is $d(t)$.

The task is to compute such a tree of minimum rectilinear length. During this paper the depth of a terminal t is always the number of internal vertices on the unique r-t-path.

Note that vertices of a feasible tree might be placed on the same position. Moreover, in an optimal solution it is possible to have edges that cross or run parallel on top of each other, which is not possible in an optimal Steiner arborescence without depth-restrictions.

1.2 Motivation and previous work

The problem is motivated by an application in VLSI design. In the repeater tree problem a signal has to be distributed from a source/root to several sinks placed on a chip by a tree-like network A consisting of vertical and horizontal wires. The signal is delayed on its way from the source s to each sink t, where the delay can be approximately measured as the length of the unique s-t-path in A plus a constant times the number of internal vertices of the path. Moreover, for every sink the signal has to arrive before a given individual arrival time. If we choose these arrival times as small as possible such that a feasible repeater tree still exists, this problem is equivalent to the problem studied in this paper: For each sink t, the source-t-path must be a shortest path and the number of vertices on this path is given by the difference of its length and the arrival times. See [1] for further details.

The minimum depth-restricted rectilinear Steiner arborescence problem is closely related to Steiner trees and Steiner arborescences. Hwang [6] proved that the classical rectilinear Steiner tree problem is NP-hard. In the rectilinear Steiner arborescence problem the task is to compute a rectilinear Steiner tree for a given set of terminals and a distinguished root such that all root-terminal-paths are shortest paths. Computing a shortest minimum rectilinear Steiner arborescence is NP-complete (see Shi and Su [8]).

Our problem is an even more restricted version of Steiner arborescences. We do not only require that all root-terminal-paths are shortest ones, but additionally the depth of each treminal is given. We prove that even this restricted version of the problem is NP-complete.

Figure 1 shows examples for minimum Steiner trees, Steiner arborescences and depth-restricted Steiner arborescences in the rectilinear plane.

Our trees can be interpreted as shallow-light Steiner arborescences with vertex delays [4]. Our results imply that computing such arborescences in the rectilinear plane is NP-hard.
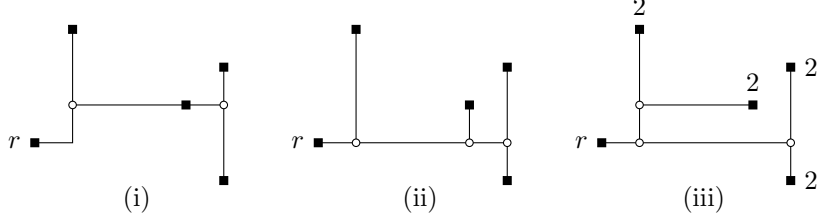

Figure 1: A shortest rectilinear Steiner tree (i), a shortest rectilinear Steiner arborescence (ii) and a shortest depth-restricted rectilinear Steiner arborescence (iii). The root of the instances is denoted by r and the numbers in (iii) denote the given depths of the terminals.

2 Feasibility

It is easy to verify, whether there exists a feasible solution for a given instance (T, p, d). To this end, let (A, π) be a feasible solution, that is, A is a binary tree where each terminal $t \in T$ is at depth $d(t)$.

By Kraft's inequality [7], there exists a binary tree with leaves at depths exactly $d_1, \ldots, d_n$ (in any order) if and only if

$$\sum_{i=1}^{n} 2^{-d_i} = 1. \tag{1}$$

Hence, for a given instance (T, p, d) a feasible solution exists if and only if (1) is satisfied for $d_i = d(v_i)$, $i \in \{1, \ldots, n\}$. In this case, it is easy to construct a feasible - but not necessarily shortest - depth-restricted Steiner arborescence: Using Huffman coding [5] one can compute a binary tree satisfying (1). Placing all internal vertices of this tree at the position of the root results in a feasible tree. We conclude that deciding whether a feasible tree exists for a given instance can be done in polynomial time. However, we are interested in the complexity of computing a shortest depth restricted tree.

3 Main Idea

In the remainder of this paper we only consider instances where the root is placed at the origin and all terminals are placed in the first quadrant. Note that this is a further restriction of the problem. It will simplify our analysis significantly. Thus in any feasible solution the parent w of a vertex v always satisfies $p_x(w) \le p_x(v)$ and $p_y(w) \le p_y(v)$.

Let A be a feasible arborescence for an instance (T, p, d). Then obviously the depth of an internal vertex v is one smaller than the depth of its two children w_1 and w_2, that is $d(v) = d(w_1) - 1 (= d(w_2) - 1)$. We extend the definition of depth to edges by setting the depth of an edge (v, w) to be $d(w)$. Two vertices can have a common parent if and only if they have to be at the same depth.

If the arborescence A is given, the optimal positions of the internal vertices can be easily computed:

Proposition 3.1 *If an internal vertex v has two children at positions (x_1, y_1) and (x_2, y_2), respectively, then the optimal position for v is $(\min\{x_1, x_2\}, \min\{y_1, y_2\})$.*

Proof: In a feasible solution we have $p_x(v) \leq \min\{x_1, x_2\}$ and $p_y(v) \leq \min\{y_1, y_2\}$. If one of the inequalities is not satisfied with equality, moving the vertex to the right or above yields another feasible embedding that is shorter. $\square$

An intermediate consequence of Proposition 3.1 is that in every optimal solution all vertices are placed on the vertices of the Hanan grid on S; that is, for every internal vertex at position (x, y) there exist two terminals with x- and y-coordinate x and y, respectively (see [3]).

4 Reduction

4.1 Reduction Overview

We prove the NP-completeness by a reduction from *Maximum 2-Satisfiability* (in short *Max-2-Sat*). A Max-2-Sat instance consists of a set of variables $\mathcal{V} = \{x_1, \ldots, x_n\}$ and a set of clauses $\mathcal{C} = \{C_1, \ldots, C_m\}$ on $\mathcal{V}$ with $|C_i| = 2$ for $i \in \{1, \ldots, m\}$. The problem is to find a truth assignment π such that the number of clauses satisfied by π is maximized. Garey et al. [2] proved that Max-2-Sat is NP-hard.

We use the component design technique to transform a Max-2-Sat instance $(\mathcal{V}, \mathcal{C})$ into an instance for our problem. First we give a high-level overview of this reduction. The construction consists of several types of gadgets that are placed on a uniform grid, where the root is located at the origin. For each variable and each clause, we have a variable gadget and clause gadget, respectively, that are placed on the diagonal of the grid (see Figure 2). The gadgets are connected by horizontal or vertical connections representing the literals: For every variable x_i, one connection is leaving the variable gadget to the left (corresponding to the literal x_i) and one connection is leaving to the bottom (corresponding to $\overline{x_i}$). Each clause gadget receives one connection from above and one from the right, representing the corresponding literals of the clause. In order to split a connection and to switch from horizontal to vertical connections or vice versa we add splitter gadgets. Finally, we require connections ensuring the existence of a feasible solution for the instance (marked by dashed lines in Figure 2).

Each truth assignment for $(\mathcal{V}, \mathcal{C})$ corresponds to a feasible Steiner arborescence where a truth assignment satisfying a maximal number of clauses corresponds to a shortest feasible Steiner arborescence. The length of the connections

within the variable and splitter gadgets and between them differ only slightly for different truth assignments, but the length of a connection of clause gadgets increases by a (relatively large) constant C if the clause is not satisfied by the truth assignment. Therefore, there exists a truth assignment π satisfying k of m clauses if and only if there exists a feasible Steiner arborescence of length at most $c + (m - k)C$ where c is a constant that is independent of π.
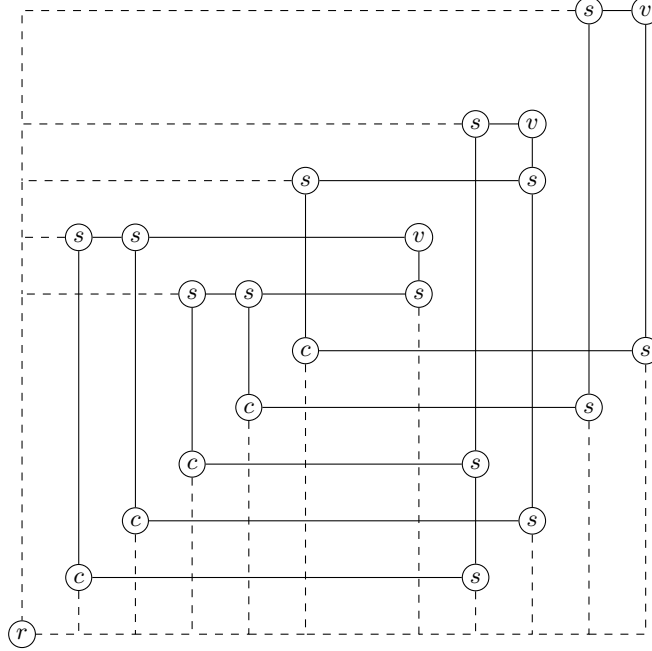

Figure 2: High-level overview of the transformed instance. Root, variable, clause and splitter gadgets are marked by r,v,c and s, respectively. The bold lines show the connections between the gadgets and the dashed lines the connections required to enable a feasible solution.

4.2   Tile design

The gadgets are realized by equal sized quadratic *tiles*. In this section we describe the design of the different types of tiles. Each tile has size $(4\alpha+2)\times(4\alpha+2)$ and contains several terminals, depending on the tile's type. The tiles are placed on a uniform grid containing $(1+m+2n)\times(1+m+2n)$ tiles and having lattice spacing $4\alpha + 2$. The integral constant α will be set later (see Section 4.8).

Figure 3 shows a prototype of a tile. The black squares show possible positions of terminals and the dotted lines show the Hanan grid on the set of possible terminal positions.

On some type of tiles we use *terminal cascades*, a set of terminals with

consecutive given depths placed at the same position. If an edge of depth $a + k$ starts at the position p of a terminal cascade containing k terminals with depths from $a + 1$ to $a + k$, then all terminals of the cascade can be connected to the tree by adding k Steiner points at position p and connecting the terminals and Steiner points appropriately. In this case, an edge of depth a ends at the cascade and we have no additional connection cost as all inserted edges have length 0. If no edge of depth $a + k$ starts at the position of the cascade, the instance is constructed such that we have k edges of length at least 1, increasing the cost to connect the cascade to the tree by at least k.

A special type of terminal cascades are the *double terminals* consisting of two terminals at the same position with consecutive depths. Double terminals are only placed on positions D_j, $j \in \{1, 2, 3, 4\}$ (see Figure 3). A tile t contains a terminal at position o_j (for $j \in \{1, 2, 3, 4\}$) if and only if there is a double terminal at position D_j. Moreover, if the double terminal at D_j, $j \in \{1, 2\}$, has depths k and $k - 1$, then o_j has depth $k - 2$ and if the double terminal at D_j, $j \in \{3, 4\}$ has depths k and $k - 1$, then o_j has depth $k + 1$.

Let (A, π) be a feasible solution, t a tile and $\pi|_t$ be the embedding we obtain by projecting all vertices that are outside of t with respect to π to the nearest point on the border of t. Then the *length of (A, π) on t* is the total length with respect to $\pi|_t$ of all edges (v, w) that contain at least one vertex in the inner of t. Note that the total length of (A, π) on all tiles is a lower bound for the length of (A, π).

Proposition 4.1 *If t is a tile with k double terminals, then every feasible solution has length at least $2k\alpha$ on t.*

Proof: Recall that double terminals are only located at positions D_j, $j \in \{1, 2, 3, 4\}$. Let H be the Hanan grid on all terminals and consider a feasible arborescence A so that all vertices are on positions of vertices of H. Let D be a double terminal at position p and P be the set of vertices of A having distance at most 1 to p. By construction, for every edge $\{v, w\}$ with $v \in P$ we have either $w \in P$ or $||p(v) - p(w)||_1 \geq \alpha$.

In a feasible arborescence at least one edge must leave P. If no edge is entering P, then it only contains the two terminals at positions p. As they have different depths, the cannot have the same parent and thus two edges must leave P. Thus $|\delta(P)| \geq 2$ implies that connecting a double terminal increases the length of a feasible connection by at least 2α. $\square$

An intermediate consequence is the following:

Conclusion 4.2 *Any feasible arborescence for an instance containing k double terminals has length at least $2k\alpha$.*

We construct the tiles and the instance I such that if I contains k double terminals, then there exists a solution A of length less than $(2k + 1)\alpha$.

Lemma 4.3 *Let (A, π) be an optimal solution of length strictly less than $(2k + 1)\alpha$ for an instance I with k double terminals. If t is a tile with terminal s*

at position o_i for some $i \in \{1, 2, 3, 4\}$, then the Steiner point connected to s is either placed at position o_i or position $\hat{o}_i$.

Proof: Denote by s' the Steiner point in A connected to s. As A is optimal, s' is placed on the Hanan grid given by the terminals. If s' is not placed at o_i or $\hat{o}_i$, then the distance between s and s' is at least α. But then the total length of A is at least $(2k+1)\alpha$, contradicting the assumption. □

During the remainder of the paper we make the following assumption on the constructed instances. By setting α to an appropriate value, we can later guarantee that the assumption is indeed satisfied.

Assumption 4.4 *If I is an instance with k double terminals, then there exists an optimal solution of cost less than $(2k+1)\alpha$.*

By Lemma 4.3 a tile t can only be entered or left at Steiner points that are connected to a terminal at position o_i, $i \in \{1, 2, 3, 4\}$. We call these Steiner points *input* or *output* of tile t if it is connected to a terminal at position o_i with $i \in \{3, 4\}$ or $i \in \{1, 2\}$, respectively. Note that each input of a tile t is the output of the tile that shares the border of t containing the input. Thus it suffices to only consider inputs in the following. For every input s, we define the depth and the *parity* of s: Consider a tile with an input connected to a terminal v at position o_i. The depth of s is $d(v) - 1$ (recall that s is a child of v). If the input is placed at the position of v, then the parity of the input is 0. Otherwise, the parity is 1. Later we associate with each input a literal. Then the literal is set to `true` if and only if the parities of the associated inputs are 1.

Now we can decompose an optimal solutions at its inputs and outputs: Consider a tile t with inputs I and parities $\pi : I \to \{0, 1\}$ for the inputs. A *tile branching* for (t, π) is a branching B (that is a forest where each tree is an arborescence) containing an arborescence for each output and the leaves of B are the terminals of t plus one leaf for each input $i \in I$ at the position corresponding to the parity $\pi(i)$. Moreover, the branching satisfies the depth-restrictions, that is, $d(w) = d(v) - 1$ for the parent w of a vertex v. Hereby we use the depths of the inputs as the depths for the terminals placed at their positions. An implication is that the depths of the roots coincide with the depths of the corresponding outputs.

A *shortest connection* for tile t with parities π for the inputs is a shortest tile branching for (t, π). Shortest connections for a given tile t with parities π can be computed efficiently by dynamic programming: We know that the Steiner points are only placed at vertices of the Hanan grid and that both children of a Steiner point must have the same depth.

4.3 Variable Tiles

For every variable of a 3-SAT instance $(\mathcal{V}, \mathcal{C})$ we build a *variable tile* containing 8 terminals and two outputs. Figure 5 shows the positions of the terminals as black squares and their depths. The dotted lines show possible positions of edges in an optimal solution.
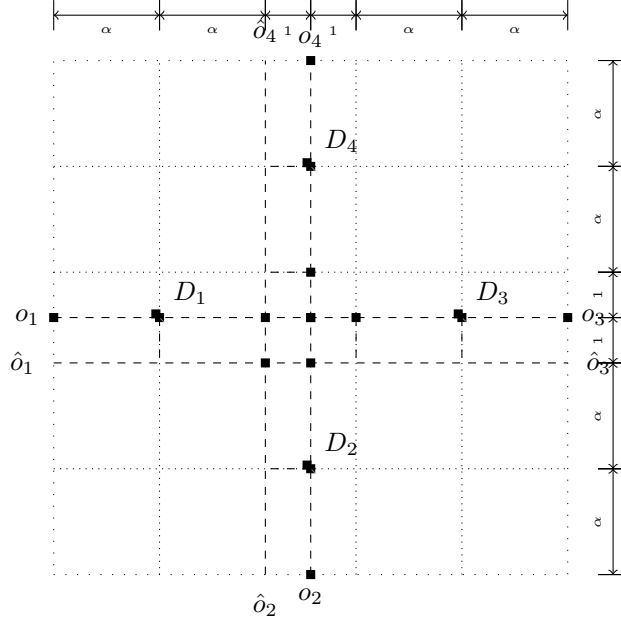
Figure 3: Prototype of a tile.

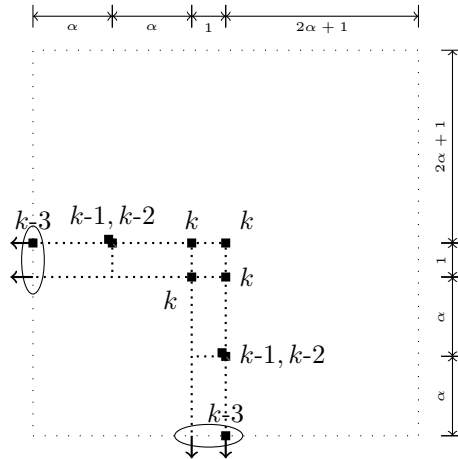

Figure 4: Explanations.

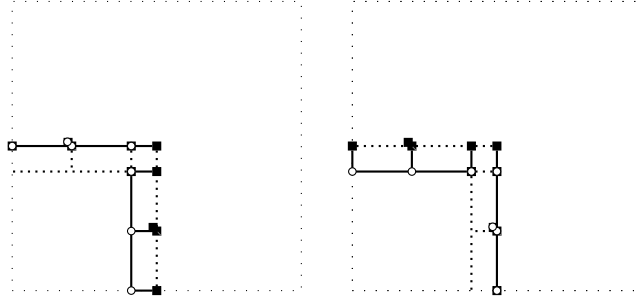Figure 5: A variable tile

8

Figure 6: The two possible shortest connections for a variable tile

Lemma 4.5 *There are two shortest connections for a variable tile. The total edge length of a minimum connection is $4\alpha + 5$.*

Proof: The two minimum connections are shown in Figure 6. Note that by the placement of the terminal at position $(2\alpha + 1, 2\alpha + 1)$ the parity of at least one of the outputs must be 0. $\qquad\square$

We associate with the output at position $o_1/\hat{o}_1$ the literal x_i and with the output at position $o_2/\hat{o}_2$ literal $\overline{x_i}$. Thereby, the connection in Figure 6 (left) corresponds to $x_i = \texttt{true}$, while the connection in Figure 6 (right) corresponds to $x_i = \texttt{false}$.

4.4 Clause Tiles

For every clause, we construct a clause tile as shown in Figure 7. It contains two terminal cascades (drawn as black rhombs), each with β terminals and both inputs have depth k. The integral value β is constant and the same for all clause tiles and will be set later. We denote the upper right terminal cascade by S_1.

The length of a minimum connection of a clause tile depends on the parities of the inputs.

Lemma 4.6 *A minimum connection for a clause tile has length $6\alpha + 9$ if both inputs have parity 1, length $6\alpha + 10$ if exactly one input has parity 1 and length $6\alpha + 11 + \beta$ if both inputs have parity 0.*

Proof: Figure 8 shows minimum connections for the four different pairs of parities of the inputs. Note that in the case where both inputs have parity 0 there are $k - l$ edges of length 1 leaving the terminal cascade at position $(2\alpha + 1, 2\alpha + 1)$. $\square$

The terminal cascade at position $(2\alpha, 2\alpha)$ enforces the parity of the output to be 0.

4.5 Connection Tiles

In oder to guarantee, that in an optimal solution the proper inputs and outputs are connected, we add *connection tiles*. A horizontal or vertical connection tile
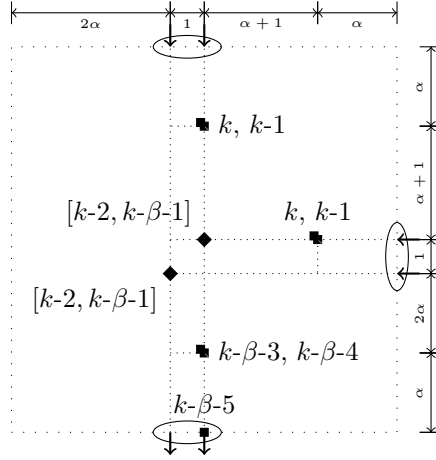
9

Figure 7: A clause tile

enables the connection of a horizontal or vertical input with an horizontal or vertical output, respectively (see Figure 9, left).

Lemma 4.7 *A minimum connection for a connection tile has length $4\alpha + 2$ if the parity of the input is 1 and $4\alpha + 8$ otherwise.*

Proof: The minimum connections are shown in Figure 10. □

Note that in a minimum connection the parities of the input and the output of a connection tile are the same.

Furthermore, we use *crossing tiles*, each consisting of the union of a vertical and an horizontal connection tile (see Figure 9, right). The vertical and horizontal connection of a crossing do not influence each other, if the depths of the corresponding terminals are distinct.

4.6 Splitter tiles

Note that a connection tile can only connect inputs and outputs that are both at horizontal or both on vertical borders of their tiles and the tiles that are connected have to be in the same row or column of the underlying grid. But there exist cases where we have to connect one input with two outputs or where the input is on a horizontal border and the output is on a vertical borer or vise versa. In these cases we "split" the path connecting inputs and outputs.

To this end, we introduce *splitter tiles* (see Figure 11). A splitter tile contains one input with depth k and two outputs. It is designed in such a way that in an optimal solution the parities of all inputs and outputs are the same. There are two types of splitter tiles; one, where the input is at the upper border and one where the input is on the right border. As these types of tiles are the same up to symmetry, we restrict ourselves to define and analyze splitter tiles where
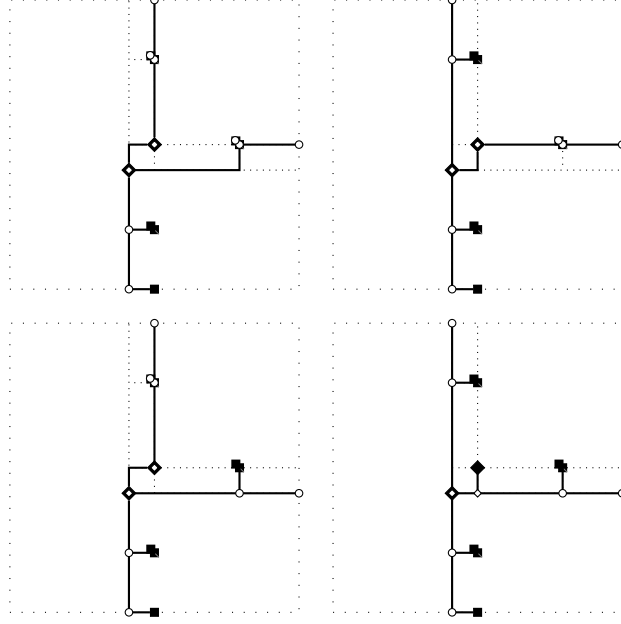
Figure 8: Shortest connections for a clause tile

the input is on the right border of the tile. The terminal cascade plays a crucial role here.

Proposition 4.8 *Each feasible connection of a splitter tile with a terminal cascade containing γ terminals has length at least $6\alpha + \gamma$.*

Proof: Consider an horizontal splitter tile t and denote by C the terminals of the terminal cascade. As t contains one input and two outputs, the induced length is at least 6α by Prop. 4.1. Let S be the set of Steiner points that are placed on the position of the terminal cascade. If S is empty, then the distance between each terminal of C and its parent is at least 1, thus the total length of the connection is at least $6\alpha + |C| = 6\alpha + \gamma$.

If on the other hand $S \neq \emptyset$, then let $s \in S$ be a vertex with highest depths. Consider the subtree rooted at s. As all terminals of C have distinct depths and by the observation that the double terminals of the input cannot be in the subtree, there must be a vertex in the subtree outside the tile. But then the induced length of t is at least $8\alpha > 4\alpha + \gamma$, where 6α comes from the double terminals and 2α from the subtree rooted at s. $\qquad\square$

Now we analyze the length of shortest connections for a splitter tile:

Lemma 4.9 *Let A be a minimum connection for a splitter with a terminal cascade containing γ terminals and set $L = 6\alpha + \gamma + 3$. If the parities of the inputs and outputs are 1, then A has length L. If the parities of the inputs and*
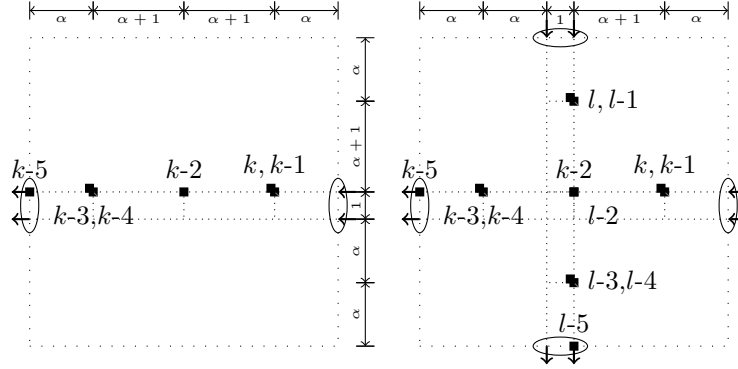
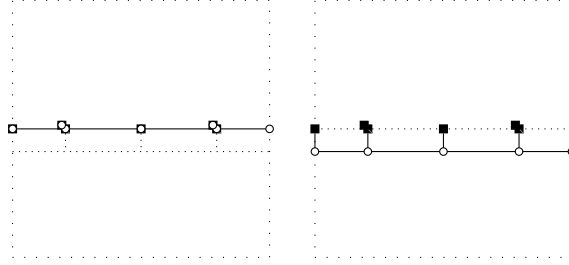Figure 9: A horizontal connection tile (left) and a crossing tile (right).

Figure 10: Shortest connections for an horizontal connection tile.

outputs are 0, *then A has length* $L + 8$. *Finally, if the parity of the input is* 0 *and the parities of the outputs are* 1 *then A has length* $L + 1 + 2\gamma$.

Proof: Figure 12 shows the three possible shortest connections. In the first two cases each terminal of the terminal cascade has to be connected to the tree by an edge of length 1, but in the last case, they are connected by edges of length 2. □

In the third case of Lemma 4.9 the parity switches from 0 at the input to 1 at the outputs, that is, the corresponding literal switches from `false` to `true`. We call this type of connection a *forbidden connection*. As this is not allowed we have to ensure that such connections are never a part of an optimal solution. So we assume that the following assumption is satisfied during the remainder of the paper.

Assumption 4.10 *If I is an instance with k double terminals and l splitter tiles with terminal cascades containing* $\gamma_1, \ldots, \gamma_l$ *terminals, respectively, then an optimal solution hast cost less than* $2k\alpha + \sum_{i \in \{1,\ldots,l\}} \gamma_i + \min_{i \in \{1,\ldots,l\}} \gamma_i$.

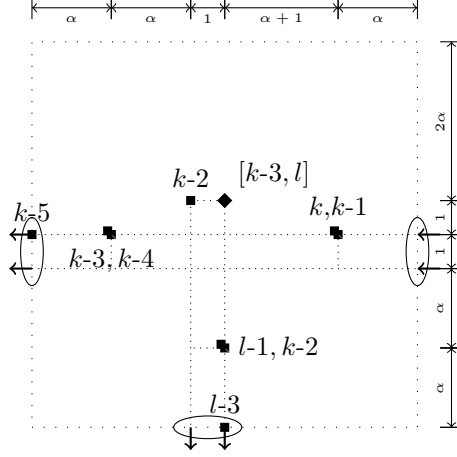If this assumption is satisfied, then the solution does not contain forbidden connections.

12

Figure 11: Horizontal splitter tile: position of the terminals and the terminal cascade and their depths.

4.7 Additional connections

Using the variable, clause, splitter and connection tiles we are now able to build the main part of our instance. But there are still some "open" outputs that are not connected yet, for example the outputs of the clauses. Thus we have to add terminals with consecutive depths in order to connect these outputs to the root r. For every additional input and output that we use we add a double terminal in order to coincide with the structure of the prototype tile. If two such connections meet, we add a terminal cascade such that these connections can merge into a single vertex. Finally, we add a terminal cascade at the root in order to guarantee the existence of a feasible solution. Note that the minimum total length of the edges required to connect these additional terminals is always the same.

4.8 Realizations of truth assignments and NP-completeness

Let T be the set of all tiles. For each tile $t \in T$ we denote by $L(t)$ the minimum length of a shortest connection for t. Then $L := \sum_{t \in T} L(t)$ is a lower bound for the length of a feasible solution.

Let $\pi : \{x_1, \ldots, x_n\} \to \{0, 1\}$ be a truth assignment. We construct a feasible solution by setting the parities of the variable tiles according to the truth assignment, inducing the parities of all inputs and outputs of the tiles and call the resulting arborescence a *realization* of π.

Figure 13 shows a transformed instance for the *Max-2-Sa* instance $(\mathcal{V}, \mathcal{C})$ defined by $V = \{x_1, x_2, x_3\}$, $\mathcal{C} = \{C_1, \ldots, C_5\}$, $C_1 = \{x_1, x_2\}$, $C_2 = \{x_1, \overline{x_2}\}$, $C_3 = \{\overline{x_1}, x_2\}$, $C_4 = \{\overline{x_1}, x_3\}$ and $C_5 = \{\overline{x_2}, \overline{x_3}\}$. Moreover, a shortest solution cor-
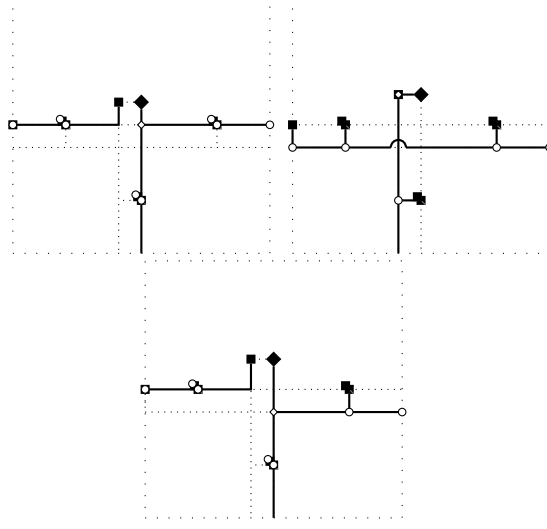
Figure 12: The three possible shortest connections for a splitter tile.

responding to the truth assignment $x_1 = x_3 = \mathtt{true}$ and $x_2 = \mathtt{false}$ is shown. The given depths have been omitted for clarity of presentation. The figure also illustrates, how the open outputs are connected to the root.

Next, we analyze the length of a realization. As seen in Lemma 4.6 the induced connection of a clause tile has length $6\alpha + 9$ or $6\alpha + 10$ if the clause is satisfied under π and $6\alpha + 11 + \beta$ otherwise. For all other tiles, the length of the induced connection is at most 8 units longer than a minimum connection for that tile. Let u be the number of clauses that are not satisfied under π. Then the length $L(\pi)$ of the solution is at least $L + 2u\beta$ and at most $L + 2u\beta + 3N^2 < L + 3u\beta$.

For the length ℓ of a realization satisfying $n - u$ of the clauses, we have

$$\ell \in [L + u\beta, L + u\beta + 10(nm)^2] =: B_u. \tag{2}$$

Now the values for α, β and γ can be specified. We have to set β such that the length of a shortest realization indicates, how many clauses are satisfied. To this end, the sets B_u, $u \in \{1, \ldots, m\}$, have to be distinct and we set

$$\beta = 20(nm)^2. \tag{3}$$

In order to satisfy Assumption 4.10, we observe that every feasible realization has length at most

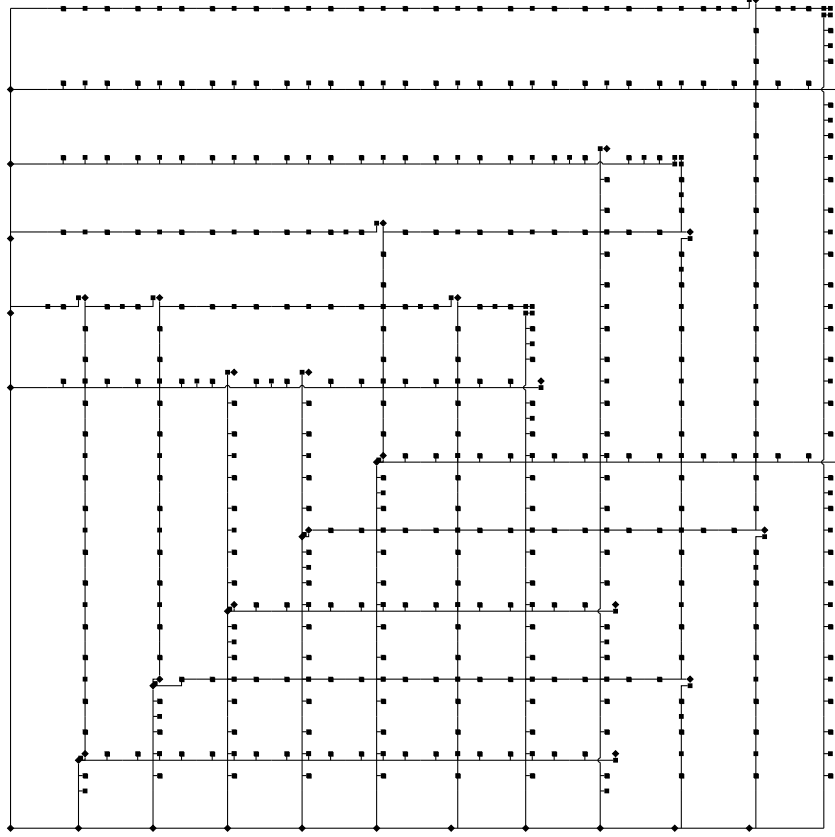$$2l\alpha + \sum_{i \in \{1, \ldots, l\}} \gamma_i + 10(nm)^2 + m\beta, \tag{4}$$

14

Figure 13: An instance of the rectilinear Steiner arborescence with depth restriction problem and a shortest solution for it.

where γ_i is the number of terminals in the terminal cascade of the i's splitter tile. Thus Assumption 4.10 is satisfied if

$$4(nm)^3 > \gamma_i > (nm)^3 \tag{5}$$

for all $i \in \{1, \ldots, l\}$. This can be achieved by setting the depths of the four terminals of each variable tile to $4(nm)^3$ and the depths of the connection and splitter tiles appropriately. This is possible as every root-terminal-path passes at most two splitter tiles.

Using (4) in (5), we conclude, that the length of a realization for an instance with k double terminals is at most

$$2k\alpha + 4l(nm)^3 + 10(nm)^2 + m20(nm)^2 \tag{6}$$

which is at most $2k\alpha + (nm)^4$ if nm is sufficiently large. By setting $\alpha := (nm)^4$ Assumption 4.4 is satisfied. Note that the values for α, β and γ and the number

15

of terminals are polynomially bounded in $n + m$. Thus we have a polynomial transformation.

All the previous observations together give us the main result of this paper.

Theorem 4.11 *The depth-restricted rectilinear Steiner arborescence problem is strongly NP-complete.*

Proof: The problem is obviously in NP. Using the transformation described in this paper we can transform a *Max-2-Sat* instance $(\mathcal{V}, \mathcal{C})$ into an instance I of the depth-restricted rectilinear Steiner arborescence problem. As the number of terminals, the depths and the distances in T are polynomially bounded in $|\mathcal{V}| + |\mathcal{C}|$, the transformation is a polynomial one. We conclude that as *Max-2-Sat* is strongly NP-complete, so is the depth-restricted rectilinear Steiner arborescence problem. □

References

[1] Christoph Bartoschek, Stephan Held, Jens Maßberg, Dieter Rautenbach, and Jens Vygen. The repeater tree construction problem. *Information Processing Letters*, 110(24):1079–1083, 2010.

[2] M. Garey, D. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1:237–267, 1976.

[3] M. Hanan. On Steiner's problem with rectilinear distance. *SIAM Journal on Applied Mathematics*, 14:255–265, 1966.

[4] S. Held and S. Rotter. Shallow-light Steiner arborescences with vertex delays. *Integer Programming and Combinatorial Optimization*, pages 229–241, 2013.

[5] D.A. Huffman. A method for the construction of minimum-redundancy codes. In *Proceedings of the IRE*, volume 40:9, pages 1098–1101, 1952.

[6] F.K. Hwang. On Steiner minimal trees with rectilinear distance. *SIAM Journal on Applied Mathematics*, 30(1):104–114, 1976.

[7] L.G. Kraft. A device for quantizing, grouping, and coding amplitude modulated pulses. Master's thesis, MIT, Cambridge, 1949.

[8] W. Shi and C. Su. The rectilinear Steiner arborescence problem is NP-complete. *SIAM Journal on Computation*, 35(3):729–740, 2005.